Package 'PIGShift'

December 7, 2015

Title Polygenic Inverse Gamma Shifts

Description Fits models of gene expression evolution to expression data from coregulated groups of genes, assuming inverse gamma distributed rate variation.

Version 1.0.1

Author Joshua G. Schraiber <schraib@uw.edu>

Maintainer Joshua G. Schraiber <schraib@uw.edu>

Imports ape, mvtnorm

License GPL-3

Depends R (>= 2.10)

Repository CRAN

NeedsCompilation no

Date/Publication 2015-12-07 23:15:42

R topics documented:

`compute.sqrt.dist` *Compute the vector of branch scaling parameters*

Description

This function computes the square root of the phylogenetic distance between each species in the tree and one other specified species. Assuming the true model is Brownian motion with no rate shifts, the distributions of trait change from a given species to any other species should be identical when divided by the square root of the distance between the two species.

Usage

```
compute.sqrt.dist(phy, norm = 1, species = phy$tip.label[-norm])
```

Arguments

`phy`	ape format phylogeny
`norm`	the species to which distances are computed
`species`	a vector of species names for which to compute distances

Value

A vector of square root of the distance between each species and norm

---

`dmvnorminvgamma` *Compute the pdf for multi-variate normal-inverse-gamma random variates*

Description

This function returns the multi-variate normal-inverse-gamma density evaluated at specific points

Usage

```
dmvnorminvgamma(x, mu, alpha, beta, T, log = FALSE)
```

Arguments

x	a matrix where each row is a sample and each column is a dimension.
mu	a vector indicating the mean in each dimension
alpha	shape parameter of the inverse gamma distribution
beta	scale parameter of the inverse gamma distribution
T	the variance-covariance matrix
log	a logical indicated whether to return the log of the pdf

Value

A vector densities corresponding to the rows of x

---

dnorminvgamma *Compute the pdf for normal-inverse-gamma random variates*

Description

This function returns the normal-inverse-gamma density evaluated at specific points

Usage

```
dnorminvgamma(x, alpha, beta, log = FALSE)
```

Arguments

x	a vector of points at which to evaluat the density
alpha	shape parameter of the inverse gamma distribution
beta	scale parameter of the inverse gamma distribution
log	a logical indicated whether to return the log of the pdf

Value

A vector densities corresponding to the entries of x

GO.groups *GO terms and genes*

Description

This dataset contains all Gene Ontology (GO) terms and their associated genes, derived from S. cerevisiae annotations.

Usage

```
data(yeast)
```

Format

A list with 3837 entries.

Source

The Gene Ontology Consortium. Gene ontology: tool for the unification of biology. Nat. Genet.. May 2000;25(1):25-9.

good.groups *Find groups represented in the data*

Description

This function takes in the list of genes for which gene expression data are available as well as the list of gene groups produced by read.groups and a minimum size (min_size) and returns those genes that have at least min_size genes with data available

Usage

```
good.groups(genes, groups, min_size = 2)
```

Arguments

genes	a vector containing the names of each gene for which expression data is available
groups	a list of gene groups, in the same format as the output of read.groups
min_size	the minimum size of groups to be considered

Value

A vector of group names that had at least min_size genes represented in the data

Examples

```
data(yeast)
length(GO.groups)
GO.groups.pruned = good.groups(colnames(yeast.hybrid),GO.groups,10)
length(GO.groups.pruned)
```

`mean_invgamma` *The mean of the inverse gamma distribution*

Description

This function returns the mean of inverse gamma distributed random variables

Usage

```
mean_invgamma(alpha, beta)
```

Arguments

`alpha`	the shape parameter of the inverse gamma distribution
`beta`	the scale parameter of the inverse gamma distribution

Value

the mean value of the parameterized inverse gamma disribution, given by beta/(alpha-1) if alpha > 1

`normalize.vcv` *Compute the variance-covariance matrix of normalized phylogenetic data*

Description

This function takes a variance-covariance matrix corresponding to some model of trait evolution along a phylogeny and returns the modified variance-covariance matrix that results from normalizing the data by the trait value in one of the species

Usage

```
normalize.vcv(vcv, which.norm = 1)
```

Arguments

`vcv`	origin variance-covariance matrix
`which.norm`	the species by which the data are normalized

Value

a matrix with nrow = ncol = ncol(vcv).

`norminvgamma.shift.like.norm` *Calculate the likelihood of normalized comparative data as Brownian motions with inverse gamma distributed rates*

Description

This function calculates the likelihood of the observed trait data assuming that each trait evolves according to an independent Brownian motion with inverse gamma distributed rates. The data are normalized relative to the trait values in a specified species.

Usage

```
norminvgamma.shift.like.norm(phy, dat, alpha, beta, rates, norm = 1)
```

Arguments

`phy`	an ape format phylogeny on which to simulate
`dat`	a matrix of comparative data, in which rows correspond to species and columns correspond to traits
`alpha`	the shape parameter of the inverse gamma distribution
`beta`	the scale parameter of the inverse gamma distribution
`rates`	a vector of rates for each branch of the phylogeny. The order of elements in rates shoud correspond to the order of phy$branches
`norm`	the species by which all the data is normalized

Value

A vector, with the likelihood of each gene the observed data

Examples

```
data(yeast)
sim.dat = norminvgamma.shift.sim.group(yeast.tree,2,2,rep(1,6),10)
norminvgamma.shift.like.norm(yeast.tree,sim.dat,2,2,rep(1,6))
```

norminvgamma.shift.sim.group

Simulate phylogenetic comparative data as Brownian motions with inverse gamma distributed rates

Description

This function simulates the evolution of a group of traits evolving as independent Brownian motions with inverse gamma distributed rates.

Usage

```
norminvgamma.shift.sim.group(phy, alpha, beta, rates, n)
```

Arguments

phy	an ape format phylogeny on which to simulate
alpha	the shape parameter of the inverse gamma distribution
beta	the scale parameter of the inverse gamma distribution
rates	a vector of rates, with each entry corresponding to an edge of phy. Rates should be in the same order as edges in phy$edge
n	the number of traits to simulate

Value

A matrix with each row corresponding to a species and each column corresponding to a trait

Examples

```
data(yeast)
norminvgamma.shift.sim.group(yeast.tree,2,2,rep(1,6),10)
```

OU.invgamma.like.norm

Calculate the likelihood of normalized comparative data as OUs with inverse gamma distributed rates

Description

This function calculates the likelihood of the observed trait data assuming that each trait evolves according to an independent Ornstein Ulenbeck processes with inverse gamma distributed rates. The data are normalized relative to the trait values in a specified species.

Usage

```
OU.invgamma.like.norm(phy, dat, alpha, beta, theta, norm = 1)
```

Arguments

phy	an ape format phylogeny on which to simulate
dat	a matrix of comparative data, in which rows correspond to species and columns correspond to traits
alpha	the shape parameter of the inverse gamma distribution
beta	the scale parameter of the inverse gamma distribution
theta	the constraint parameter of the Ornstein-Uhlenbeck process
norm	the species by which all the data is normalized

Value

A vector, with the likelihood of each gene the observed data

Examples

```
data(yeast)
sim.dat = OU.invgamma.sim.group(yeast.tree,2,2,2,10)
OU.invgamma.like.norm(yeast.tree,sim.dat,2,2,2)
```

OU.invgamma.sim.group *Simulate phylogenetic comparative data as OUs with inverse gamma distributed rates*

Description

This function simulates the evolution of a group of traits evolving as independent Ornstein-Uhlenbeck processes with inverse gamma distributed rates.

Usage

```
OU.invgamma.sim.group(phy, theta, alpha, beta, n)
```

Arguments

phy	an ape format phylogeny on which to simulate
theta	the strength of constraint
alpha	the shape parameter of the inverse gamma distribution
beta	the scale parameter of the inverse gamma distribution
n	the number of traits to simulate

Value

A matrix with each row corresponding to a species and each column corresponding to a trait

Examples

```
data(yeast)
OU.invgamma.sim.group(yeast.tree,2,2,2,10)
```

OU.vcv *The variance covariance matrix for an Ornstein-Uhlenbeck process*

Description

This function returns the variance-covariance matrix corresponding to an Ornstein-Uhlenbeck process run along a phylogeny

Usage

```
OU.vcv(phy, theta)
```

Arguments

phy	an ape format phylogeny
theta	the constraint parameter of the Ornstein-Uhlenbeck process

Value

a matrix with nrow = ncol = length(phy$tip.label), where the i,jth entry corresponds to the covariance between species i and j. NB: this is computed assuming that the rate of evolution is equal to one, and can be rescaled simply by multiplying

plot_logfoldchange *Plot densities of expression differences, possibly normalized*

Description

This function plots the densities of log fold change in each species relative to a single species. Expression differences may be normalized, to assess the fit of the phylogentic model

Usage

```
plot_logfoldchange(dat, groups, group_name = "", normalize = rep(1,
  nrow(dat)), remove.row = 1, color = 1:nrow(dat), main = group_name,
  names.arg = rownames(dat)[-remove.row], plot.legend = T, lwd = 1)
```

Arguments

dat	a matrix of comparative data, in which rows correspond to species and columns correspond to traits
groups	a list of gene groups, in the same format as the output of read.groups
group_name	the name of the gene group for which to plot density. Leave as default to use all genes
normalize	a vector of normalizations, corresponding to the order of speices in dat

remove.row	a species of the data matrix to remove. Use this option to ensure that the normalizing species is not plotted
color	a vector of colors in which to plot densities
main	the title of the plot. Default is group name.
names.arg	the name to assign to each density. Default is species name.
plot.legend	a logical indicating whether to plot a legend
lwd	the line width for each density

Value

Nothing

Examples

```
data(yeast)
sqrt.dist = compute.sqrt.dist(yeast.tree)
par(mfrow=c(1,2))
test_group = "GO:0007346|regulation of mitotic cell cycle"
plot_logfoldchange(yeast.homozygote,GO.groups,test_group)
plot_logfoldchange(yeast.homozygote,GO.groups,test_group,normalize=sqrt.dist)
```

---

plot_wAICbarplot *Plot a barplot of AIC weights for each model and each gene group*

Description

This function will plot a barplot where each bar corresponds to a gene group and the proportion of each bar that is filled with a certain color corresponds to the AIC weight for a given model. Bars are sorted according to which model has the highest weight.

Usage

```
plot_wAICbarplot(dat, names, col = 2:(length(names) + 1), title = "",
  cex = 1.4, border = par("fg"), space = NULL, names.arg = 1:nrow(dat))
```

Arguments

dat	a wAIC matrix, rows are be gene groups and columns are models
names	a vector of names for each model
col	a vector of colors, one color corresponding to each model
title	name for the barplot
cex	character expansion factor
border	type of border around each bar in the barplots
space	amount of space between bars in the barplot
names.arg	names of bars in the barplot

Value

invisibly returns the ordering of gene groups in the barplot.

Examples

```
## Not run:
data(yeast)
GO.groups.pruned = good.groups(colnames(yeast.homozygote),GO.groups,30)
test_groups = GO.groups[GO.groups.pruned[1:100]]
yeast.test = test.groups(yeast.tree,yeast.homozygote,test_groups,print_names=T)
plot_wAICbarplot(mytest$wAIC,1:7)

## End(Not run)
```

---

read.exp *Read gene expression data from a file, sorted by a phylogenetic tree*

Description

This funciton will read in gene expression data from a text file and format it for downstream analysis. In particular, it will ensure that species are sorted appropriately given the phylogenetic tree and that the data is appropriately normalized by one species

Usage

```
read.exp(filename, phy, transpose = FALSE, normalize = 1, sep = "\t")
```

Arguments

filename	expression data file. Ideally, rows correspond to species and columns correspond to genes. First column should be species names, first row should be gene names. If the file is formatted in a transpose (i.e. genes are rows and species are columns) then see the transpose argument
phy	an ape-format phylogenetic tree containing all the species that have gene expressiond data. The tip labels of phy should correspond to the species names in the gene expression data file
transpose	a logical indicating whether to transpose the data from the input file. Only necessary if the input file has rows corresponding to genes and columns corresponding to species
normalize	indicates which species to normalize by. If normalization is undesired (unlikely) set to 0.
sep	the character that separates entries in the expression file

Value

A matrix containing gene expression data, in which rows correspond to species and columns correspond to genes

`read.groups` *Read in the members of a gene group*

Description

This function reads in the names of the members of a group of genes and stores them to a list. The input file needs to be formatted appropriately, with one group per line: groupname<tab>gene1<tab>gene2...geneK

Usage

```
read.groups(filename, sep = "\t")
```

Arguments

`filename`	the file containing the list of gene groups
`sep`	indicate the appropriate separator if not tab

Value

A list whose names are gene group names and whose elements are vectors of genes

`remove.from.genome` *Remove genes from a specified group from the data*

Description

This function will remove data corresponding to genes that are members of groups.to.remove.

Usage

```
remove.from.genome(dat, groups, groups.to.remove)
```

Arguments

`dat`	gene expression data, rows are species, columns are gene, including colnames.
`groups`	a list of gene groups, in the same format as the output of read.groups
`groups.to.remove`	a vector of group names, the members of which will be removed from dat

Value

A new matrix of data with the genes that belonged to groups.to.remove gone.

Examples

```
data(yeast)
GO.groups.pruned = good.groups(colnames(yeast.hybrid),GO.groups,10)
dim(yeast.hybrid)
to_remove = setdiff(names(GO.groups),GO.groups.pruned)
yeast.hybrid.pruned = remove.from.genome(yeast.hybrid,GO.groups,to_remove)
dim(yeast.hybrid.pruned)
```

`rnorminvgamma` *Draw normal-inverse-gamma distributed random variates*

Description

This function will return normal-inverse-gamma distributed random variates

Usage

```
rnorminvgamma(n, alpha, beta)
```

Arguments

`n`	the number of variates to generate
`alpha`	shape parameter of the inverse gamma distribution
`beta`	scale parameter of the inverse gamma distribution

Value

A vector of random variates arising form the normal-inverse-gamma distribution

---

`test.groups` *Test all possible single-rate shift Brownian motion models and an Ornstein-Uhlenbeck model for an arbitrary number of predefined gene groups.*

Description

This function will find the maximum likelihood estimate of the parameters of every single rate shift model that is compatible with the phylogeny phy, as well as the likelihood and wAIC for each model and for each gene group. The procedure is described in Schraiber et al (2013).

Usage

```
test.groups(phy, dat, groups, norm = 1, print_names = F)
```

Arguments

phy	an ape format phylogeny
dat	a matrix of gene expression data. Rows of dat correspond to species and columns of dat correspond to genes
groups	a list of gene groups, formatted like the output of read.groups
norm	the species by which data should be normalized
print_names	a logical indicating whether to print the name of the group currently being analyzed. Useful to keep track of the progress of the function

Value

A list of several elements. wAIC is a matrix of Akaike weights for each model for each group (rows are groups, columns are models), alpha is the maximum likelihood shape parameter of the inverse gamma distribution for each model and group, beta is the maximum likelihood scale parameter of the inverse gamma distribution for each model and group, and shift is the maximum likelihood rate shift parameter for each model and each group, except for the final model which is Ornstein-Uhlenbeck, in which case it corresponds to the constraint parameter. Branches indicates which branches of the tree experience a rate shift.

Examples

```
## Not run:
data(yeast)
GO.groups.pruned = good.groups(colnames(yeast.homozygote),GO.groups,30)
to_test = GO.groups[GO.groups.pruned[1:100]]
yeast.test = test.groups(yeast.tree,yeast.homozygote,to_test,print_names=T)

## End(Not run)
```

---

test.subtrees *Test all possible single-rate shift Brownian motion models and an Ornstein-Uhlenbeck model*

Description

This function will find the maximum likelihood estimate of the parameters of every single rate shift model that is compatible with the phylogeny phy, as well as the likelihood and wAIC for each model. The procedure is described in Schraiber et al (2013).

Usage

```
test.subtrees(phy, dat, norm = 1)
```

Arguments

phy	an ape format phylogeny
dat	a matrix of gene expression data. Rows of dat correspond to species and columns of dat correspond to genes
norm	the species by which data should be normalized

Value

A list of several elements: res is the full output of the optim runs used maximize the likelihood, branches are lists of the branches that have a rate shift for each model, LL is the log likelihood for each model, wAIC is the Akaike information criterion weight for each model, alpha are maximum likelihood estimates of the shape parameter of the inverse gamma distribution for each model, beta are maximum likelihood estimates of the scale paramter for each model and shift are maximum likelihood estimates of the rate shift parameter for each model (except for Ornstein-Uhlenbeck, in which shift is an estimate of the constraint parameter of the OU process).

---

var_invgamma *The variance of the inverse gamma distribution*

Description

This function returns the variance of inverse gamma distributed random variables

Usage

```
var_invgamma(alpha, beta)
```

Arguments

alpha	the shape parameter of the inverse gamma distribution
beta	the scale parameter of the inverse gamma distribution

Value

the variance of the parameterized inverse gamma disribution, given by beta^2/((alpha-1)^2*(alpha-2)) if alpha > 2

yeast.homozygote *Yeast homozygote transcription profiles*

Description

This dataset has normalized log-fold-change in RNA levels between 3 yeast species and S. cerevisiae. Generated by performing RNAseq on homozygotes of each species to quantify expression.

Usage

```
data(yeast)
```

Format

A matrix with 4 rows and 4835 columns. Each row is a species and each column is a gene.

Source

Schraiber, et al (2013). Inferring evolutionary histories of pathway regulation from transcriptional profiling data. PLoS Computational Biology 9:e10003255.

yeast.hybrid *Yeast hybrid transcription profiles*

Description

This dataset has normalized log-fold-change in RNA levels between 3 yeast species and S. cerevisiae, generated by creating hybrids of each species with S. cerevisiae and using RNAseq to quantify allele specific expression.

Usage

```
data(yeast)
```

Format

A matrix with 4 rows and 4835 columns. Each row is a species and each column is a gene.

Source

Schraiber, et al (2013). Inferring evolutionary histories of pathway regulation from transcriptional profiling data. PLoS Computational Biology 9:e10003255.

`yeast.tree` *Phylogenetic tree of yeast species*

Description

Ape-format ultrametric phylogenetic tree of the yeast species used in Schraiber, et al (2013).

Usage

```
data(yeast)
```

Format

ape tree object

Source

Schraiber, et al (2013). Inferring evolutionary histories of pathway regulation from transcriptional profiling data. PLoS Computational Biology 9:e10003255.

Index